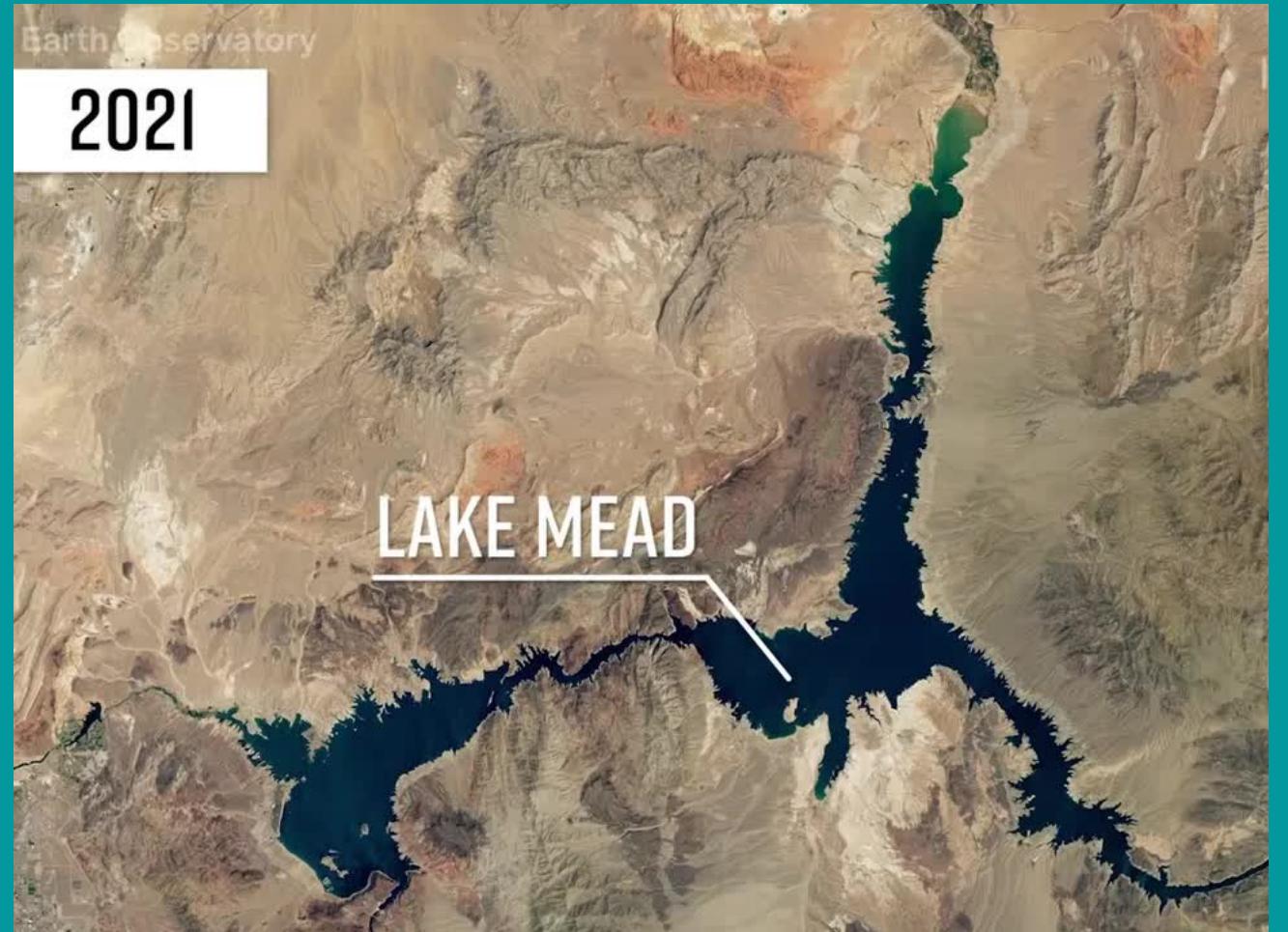
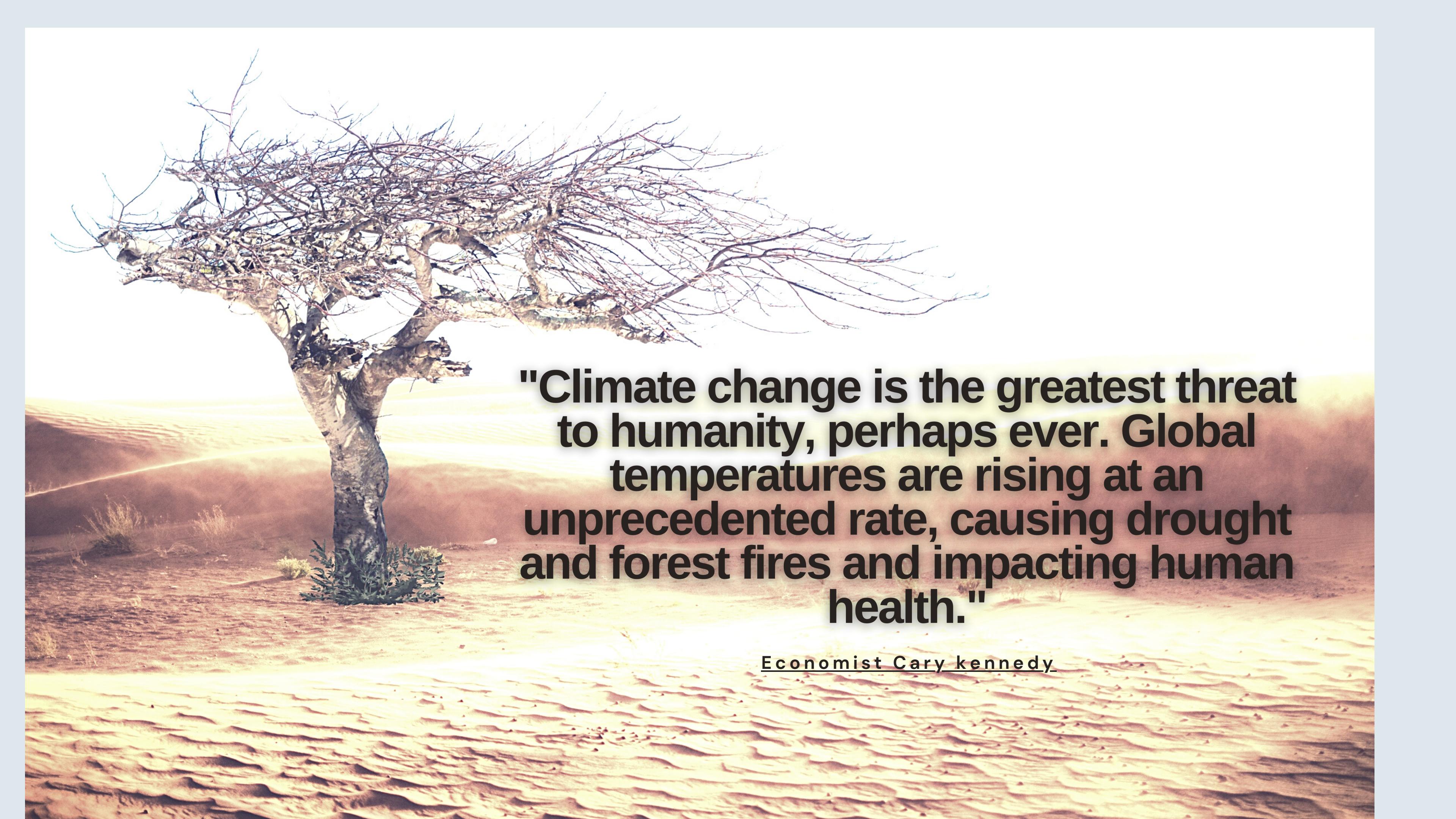


# DROUGHT PREDICTION

Machine Learning Model

FINAL PROJECT





**"Climate change is the greatest threat to humanity, perhaps ever. Global temperatures are rising at an unprecedented rate, causing drought and forest fires and impacting human health."**

Economist Cary kennedy

# What is drought?

- A PROLONGED PERIOD OF ABNORMALLY LOW RAINFALL, LEADING TO A SHORTAGE OF WATER.
- IT IS UNPREDICTABLE, COMPLEX AND NOT WELL UNDERSTOOD.

## Factors affecting occurrence and frequencies

- RAINFALL
- RELATIVE HUMIDITY
- TEMPERATURE
- WIND
- SOLAR RADIATION



# Research Scope

- To build a machine learning model that will predict the likelihood of drought occurring in the US for use in the Agricultural Sector.

## Application of the Prediction Model

To improve drought resilience by:

- making different crop choices
- enrolling in crop insurance and other farm risk management
- investing in soil health and irrigation efficiency



# Data Sources

**Kaggle Data** (<https://www.kaggle.com/datasets/cadminix/us-drought-meteorological-data>)

- Training Data(`train_timeseries.csv`) (2GB size, 2million records 2000-2016)
- Testing Data(`test_timeseries.csv`)(2019-2020)
- Geographical Data(`soil.csv`)



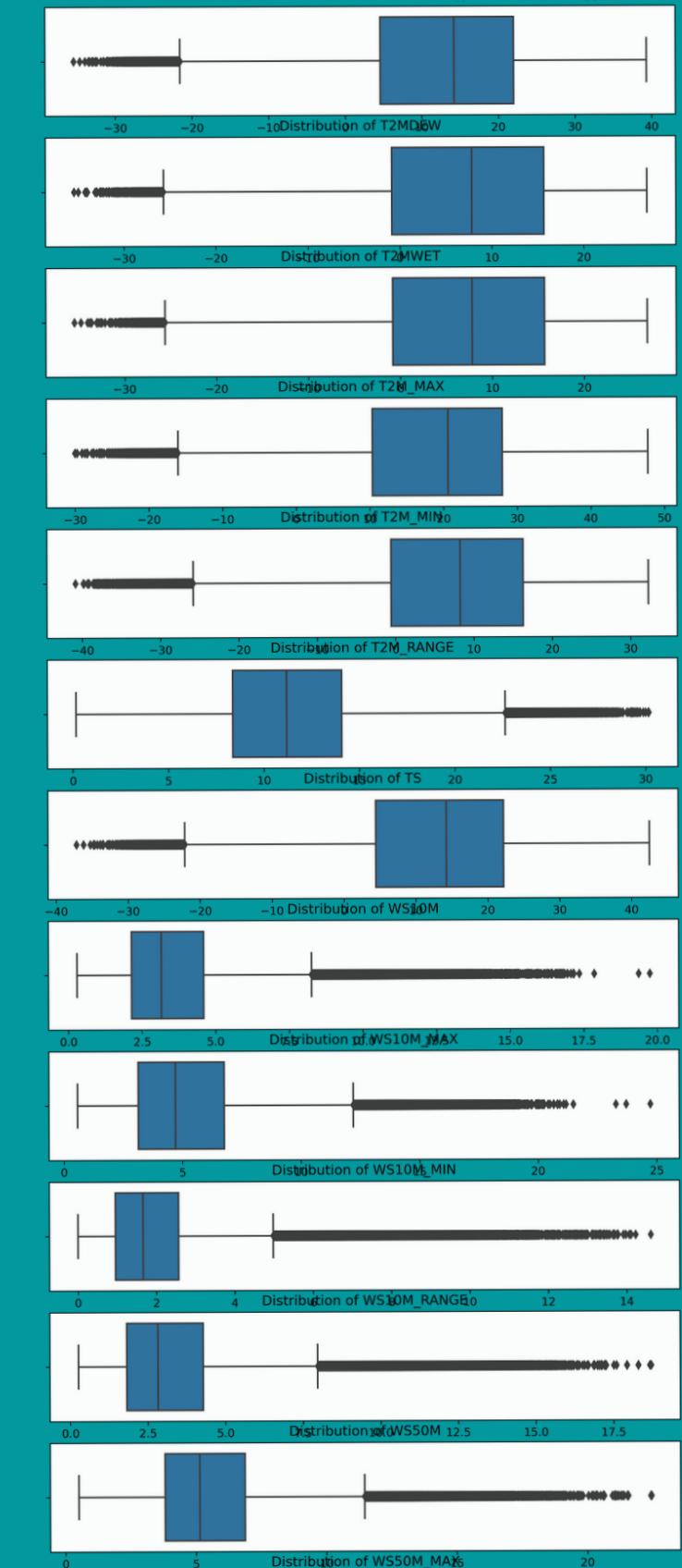
*Original Source: NASA Langley Research Center (LaRC) POWER Project funded through the NASA Earth Science/Applied Science Program*

# Data Cleanup

- Import raw data csv files to jupyter notebook
- Remove the NaN values
- Change datatype

# Data Exploration Analysis (EDA)

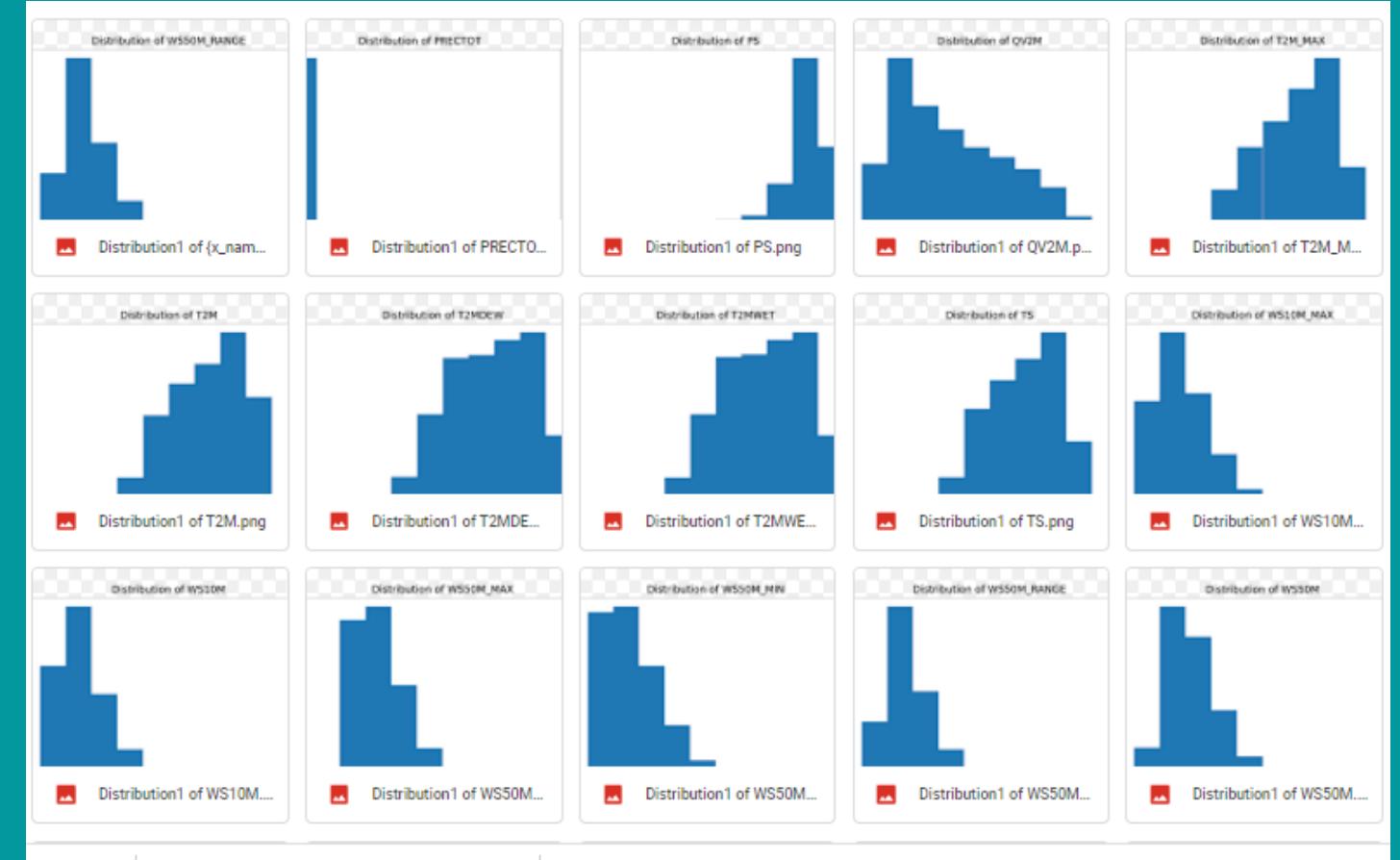
- Two methods for EDA
  - Skewness
  - Kurtosis
- Outlier Treatment and removing values beyond the standard outlier limit. First clean Data!



# UNIVARIATE ANALYSIS

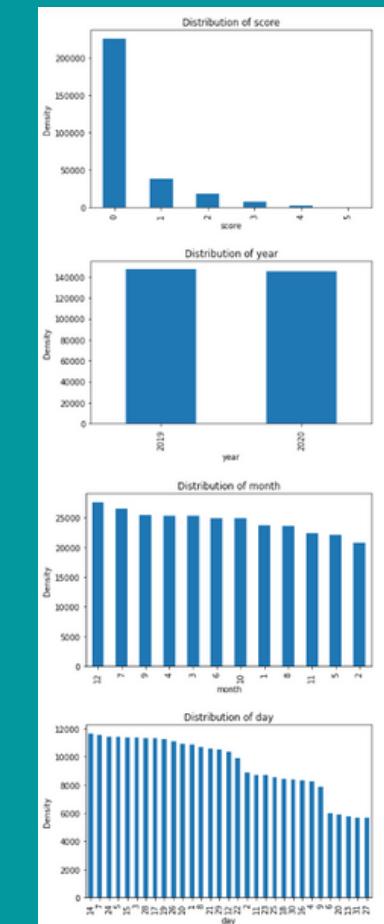
## Continuous variables (all variables)

- Normal Distribution of 21 variables by plotting Histogram



## Categorical variables(score, year, month and day)

- Normal to almost uniform distribution of 4 variables by plotting Histogram



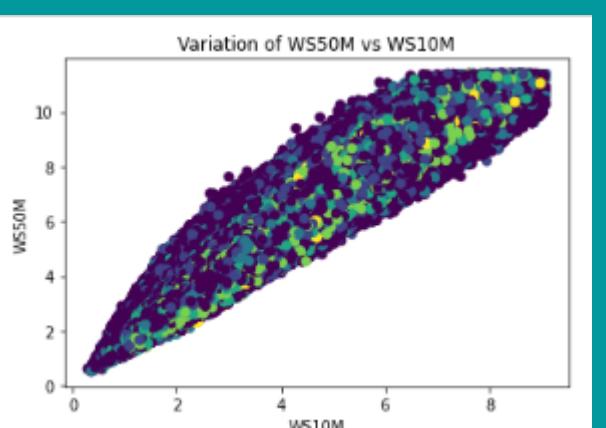
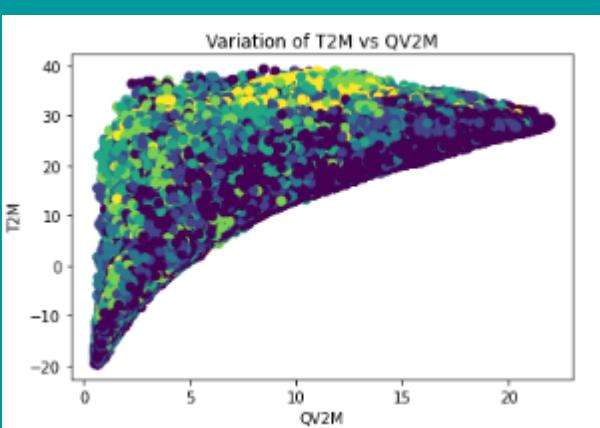
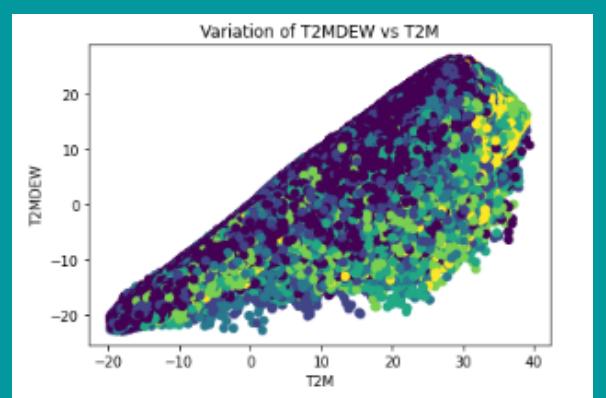
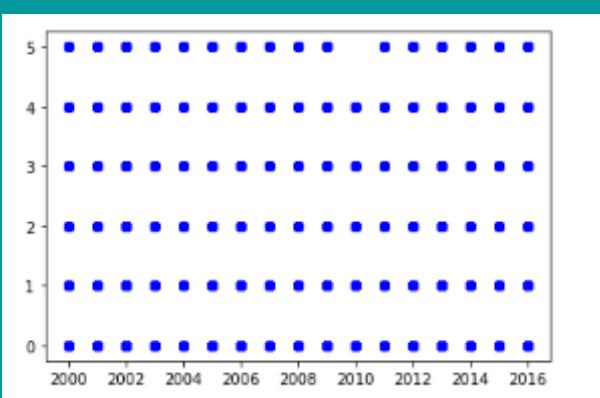
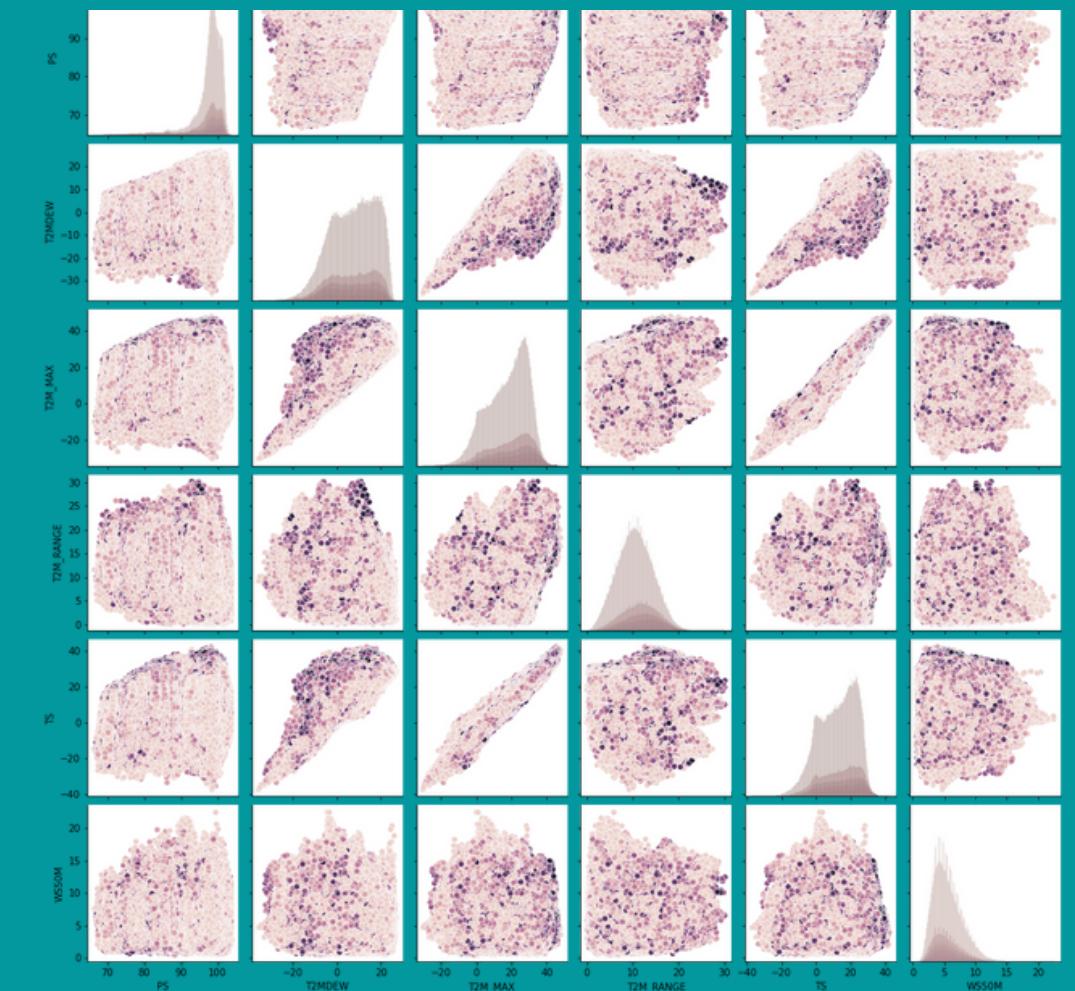
# BIVARIATE ANALYSIS

## AGRICULTURE

- Drought parameters for agriculture purpose and the impact of meteorological conditions.
- Looking for if main meteorological parameters such as precipitation, moisture, wind and temperature really contribute to ML
- 

## DATA ANALYSIS

- Used Seaborn for scatter matrix to find correlation between dependent and independent variables
- Histogram and a lot of 2D scatter plots for the distribution along the diagonal



# Correlation Heat Map

- To find correlation between independent variables for feature selection
- Values closer to 0 indicate no correlation whereas value closer to 1 indicate positive correlation between the variables

	PRECTOT	PS	QV2M	T2M	T2MDEW	T2MWET	T2M MAX	T2M MIN	T2M RANGE	TS	WS10M	WS10M MAX	WS10M MIN	WS10M RANGE	WS50M	WS50M MAX	WS50M MIN	WS50M RANGE
PRECTOT	1.000	0.069	0.245	0.093	0.231	0.231	0.027	0.145	-0.304	0.090	0.050	0.061	0.023	0.066	0.069	0.080	0.058	0.047
PS	0.069	1.000	0.282	0.164	0.341	0.341	0.112	0.208	-0.226	0.164	-0.081	-0.136	0.023	-0.198	-0.043	-0.092	0.036	-0.154
QV2M	0.245	0.282	1.000	0.870	0.959	0.960	0.804	0.906	-0.072	0.863	-0.225	-0.256	-0.109	-0.269	-0.206	-0.250	-0.082	-0.246
T2M	0.093	0.164	0.870	1.000	0.914	0.914	0.983	0.982	0.244	0.998	-0.208	-0.220	-0.125	-0.209	-0.193	-0.206	-0.113	-0.160
T2MDEW	0.231	0.341	0.959	0.914	1.000	1.000	0.855	0.940	-0.016	0.905	-0.238	-0.269	-0.116	-0.281	-0.204	-0.245	-0.082	-0.239
T2MWET	0.231	0.341	0.960	0.914	1.000	1.000	0.855	0.941	-0.016	0.906	-0.238	-0.268	-0.116	-0.280	-0.204	-0.245	-0.082	-0.239
T2M_MAX	0.027	0.112	0.804	0.983	0.855	0.855	1.000	0.938	0.408	0.980	-0.217	-0.222	-0.142	-0.200	-0.196	-0.196	-0.133	-0.126
T2M_MIN	0.145	0.208	0.906	0.982	0.940	0.941	0.938	1.000	0.065	0.979	-0.206	-0.226	-0.113	-0.225	-0.198	-0.226	-0.097	-0.200
T2M_RANGE	-0.304	-0.226	-0.072	0.244	-0.016	-0.016	0.408	0.065	1.000	0.242	-0.080	-0.043	-0.111	0.019	-0.042	0.030	-0.129	0.163
TS	0.090	0.164	0.863	0.998	0.905	0.906	0.980	0.979	0.242	1.000	-0.190	-0.203	-0.110	-0.196	-0.181	-0.193	-0.102	-0.152
WS10M	0.050	-0.081	-0.225	-0.208	-0.238	-0.238	-0.217	-0.206	-0.080	-0.190	1.000	0.952	0.833	0.703	0.966	0.909	0.795	0.412
WS10M_MAX	0.061	-0.136	-0.256	-0.220	-0.269	-0.268	-0.222	-0.226	-0.043	-0.203	0.952	1.000	0.690	0.866	0.911	0.947	0.660	0.592
WS10M_MIN	0.023	0.023	-0.109	-0.125	-0.116	-0.116	-0.142	-0.113	-0.111	-0.110	0.833	0.690	1.000	0.236	0.839	0.667	0.944	-0.046
WS10M_RANGE	0.066	-0.198	-0.269	-0.209	-0.281	-0.280	-0.200	-0.225	0.019	-0.196	0.703	0.866	0.236	1.000	0.643	0.811	0.235	0.827
WS50M	0.069	-0.043	-0.206	-0.193	-0.204	-0.204	-0.196	-0.198	-0.042	-0.181	0.966	0.911	0.839	0.643	1.000	0.918	0.848	0.374
WS50M_MAX	0.080	-0.092	-0.250	-0.206	-0.245	-0.245	-0.196	-0.226	0.030	-0.193	0.909	0.947	0.667	0.811	0.918	1.000	0.647	0.675
WS50M_MIN	0.058	0.036	-0.082	-0.113	-0.082	-0.082	-0.133	-0.097	-0.129	-0.102	0.795	0.660	0.944	0.235	0.848	0.647	1.000	-0.126
WS50M_RANGE	0.047	-0.154	-0.246	-0.160	-0.239	-0.239	-0.126	-0.200	0.163	-0.152	0.412	0.592	-0.046	0.827	0.374	0.675	-0.126	1.000

# Data Modelling

## Features Selection

- Used Random Forest Classifier model to select the features
- The features are as follows:
  1. PS(Surface Pressure)
  2. T2MDEW(Dew Point at 2M)
  3. T2M\_MAX(Maximum Temperature at 2M)
  4. T2M\_RANGE(Temperature Range at 2M)
  5. TS(Earth Skin Temperature)
  6. WS50M(Wind Speed at 50M)

```
model = RandomForestClassifier(n_estimators=10) # n_estimators is the hyperparameter
rfe = RFE(model, n_features_to_select=6) # n_features_to_select is chosen on a trial and error basis
fit = rfe.fit(X_train, y_train)
print("Num Features: %s" % (fit.n_features_))
print("Selected Features: %s" % (fit.support_))
print("Feature Ranking: %s" % (fit.ranking_))
selected_features = independent_variables.columns[(fit.get_support())]
print(selected_features)

Num Features: 6
Selected Features: [False  True False False  True False  True  True False False
 False False  True False False False False False]
Feature Ranking: [13  1   5   8   1 11  1   4   1   1 14  2 16 10  1   9 12  7   6 15  3]
Index(['PS', 'T2MDEW', 'T2M_MAX', 'T2M_RANGE', 'TS', 'WS50M'], dtype='object')
```

Random Forest Algorithm without resampling - Setting the right Hyperparameters

```
In [59]: RF_classifier = RandomForestClassifier(n_estimators = 50, max_depth=80, bootstrap=False, max_features='sqrt', random_state=0)
RF_classifier.fit(X_train, y_train)
y_pred_RF = RF_classifier.predict(X_test)

In [60]: pickle.dump(RF_classifier, open('RF_classifier.pkl', 'wb'))

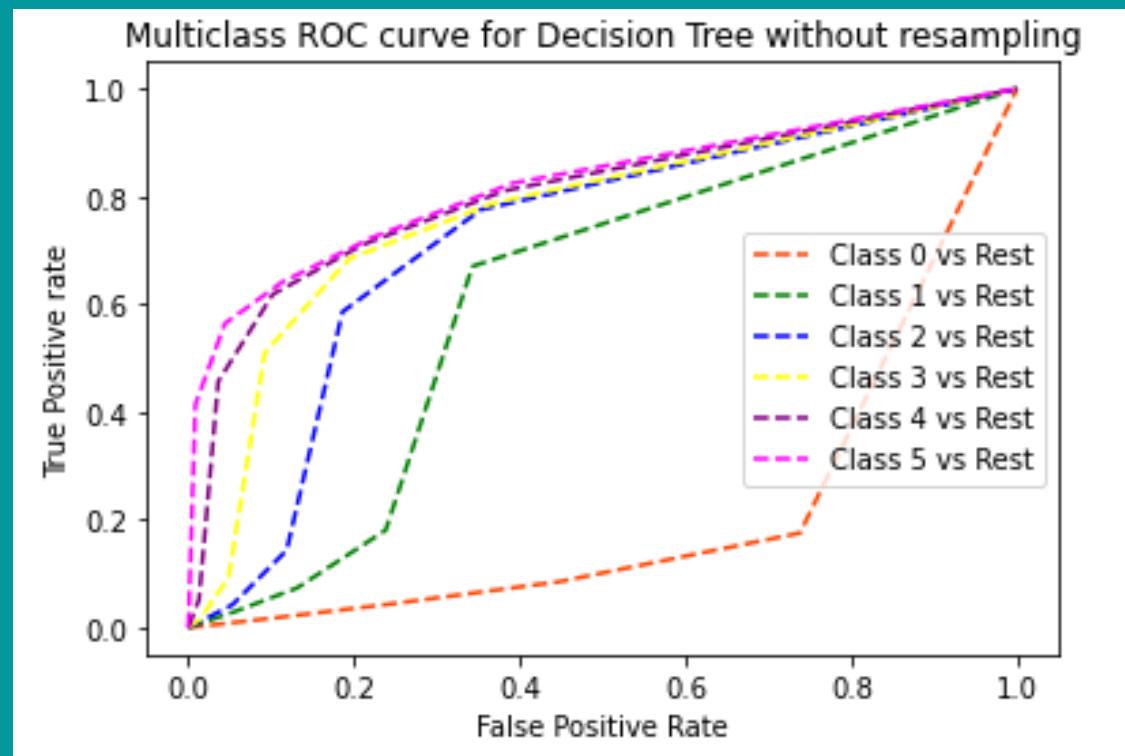
In [61]: import gzip, pickle, pickletools

In [62]: filepath = "RF_classifier.pkl"
with gzip.open(filepath, "wb") as f:
    pickled = pickle.dumps(RF_classifier)
    optimized_pickle = pickletools.optimize(pickled)
    f.write(optimized_pickle)
```

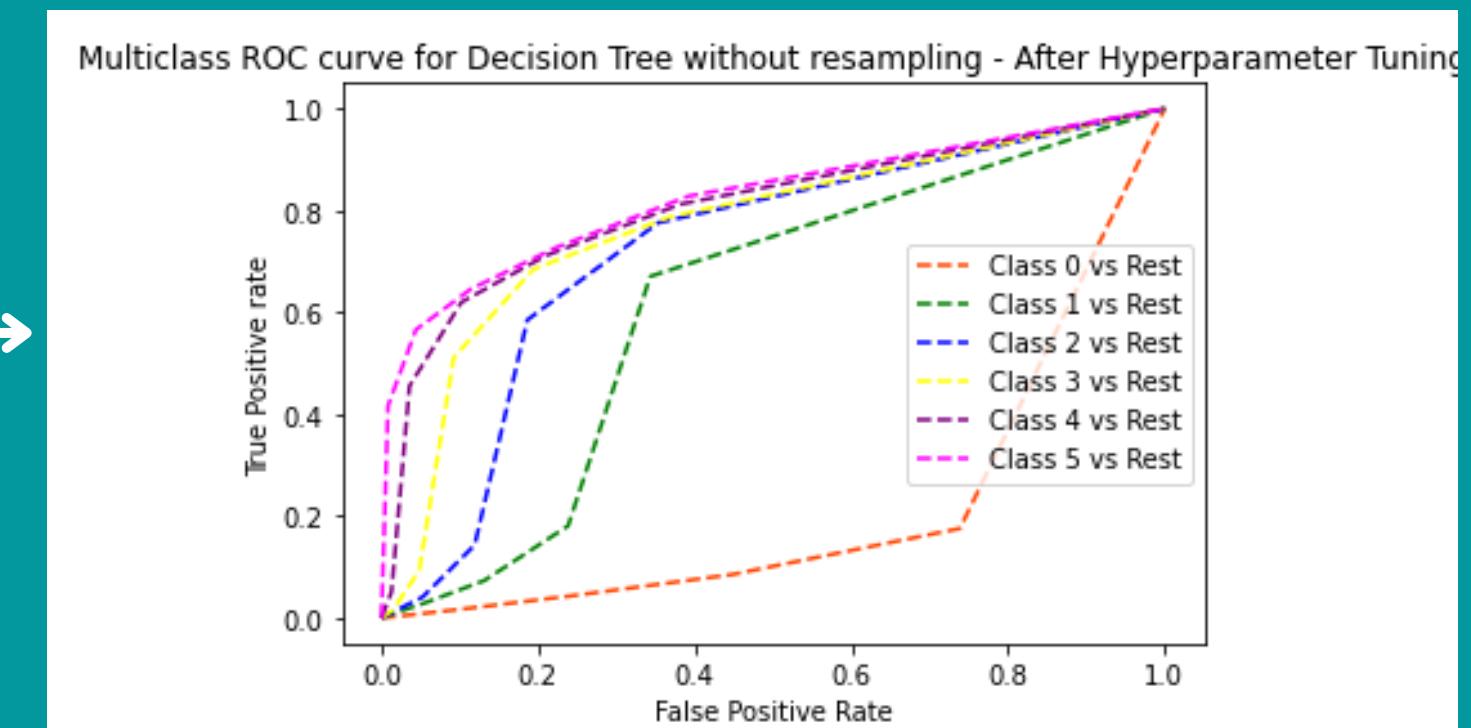
# Data Modelling

## Model Development

### Decision Tree Algorithm without resampling



After Hyperparameter tuning →



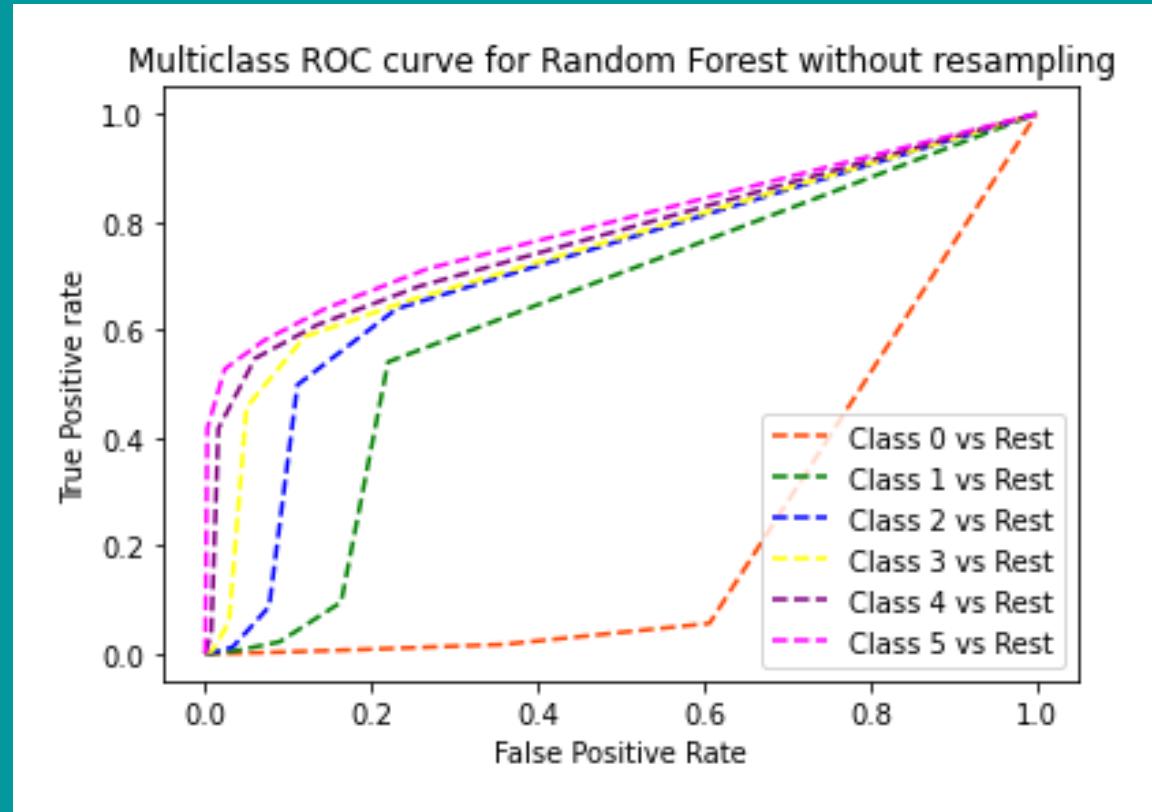
Accuracy: 0.6777989281990349  
Precision: 0.6788165543184689  
Recall: 0.6777989281990349  
F1 Score: 0.6783019240402571  
Cohen Kappa Score: 0.4536519273494525

Accuracy: 0.6778898615388346  
Precision: 0.6787938223854018  
Recall: 0.6778898615388346  
F1 Score: 0.6783368711313027  
Cohen Kappa Score: 0.4536748664726149

# Data Modelling

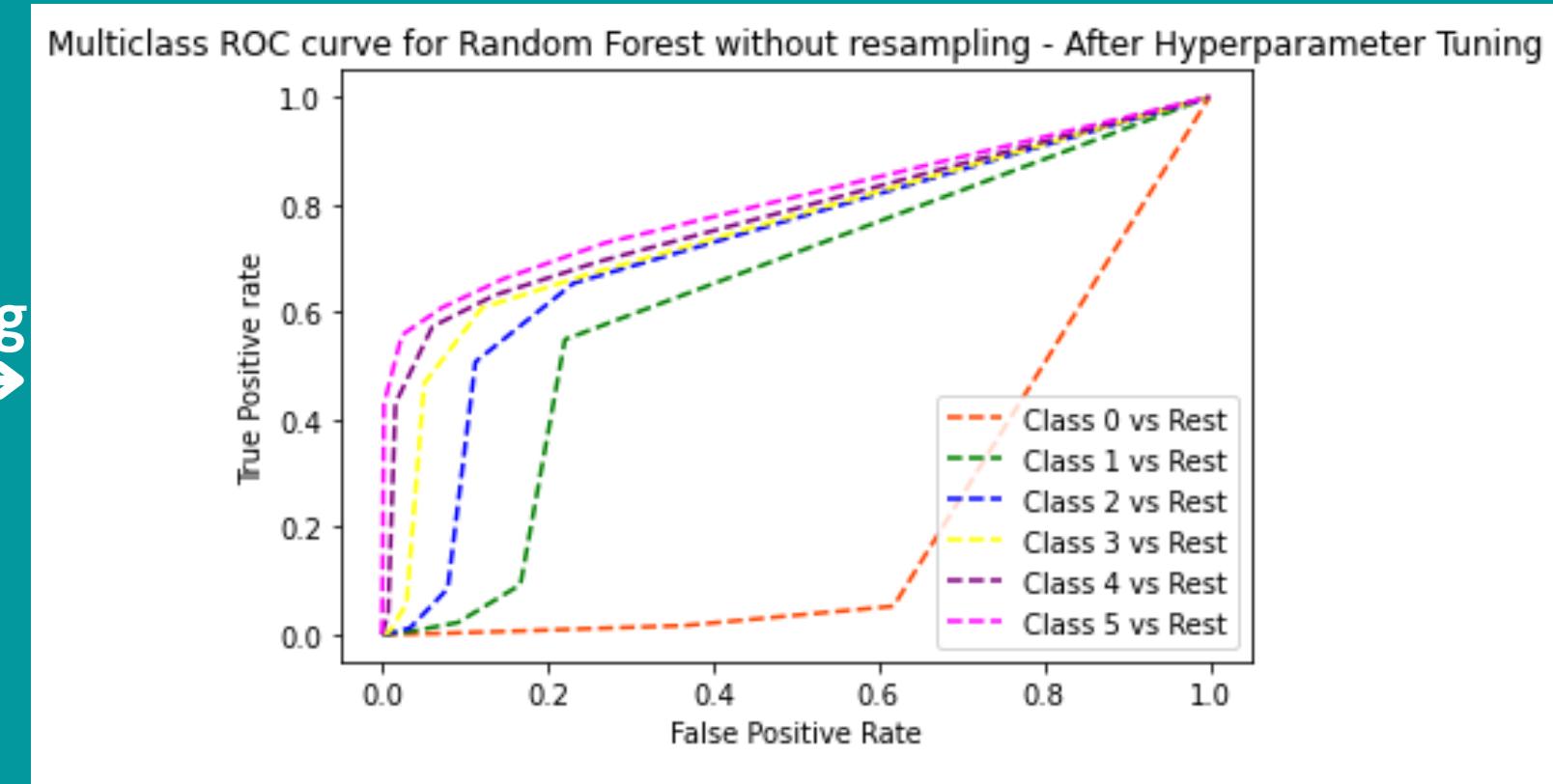
## Model Development

### Random Forest Algorithm without resampling



Accuracy: 0.7369015575870738  
Precision: 0.7154562883084277  
Recall: 0.7369015575870738  
F1 Score: 0.7148504178166684  
Cohen Kappa Score: 0.5019311370578048

Performance of RF Algorithm without resampling - After Hyperparameter Tuning:					
[[283544 10757 2663 1632 549 100] [ 37483 37892 5855 1327 410 99] [ 18215 7688 22177 3838 518 113] [ 11427 2109 4845 14137 1827 148] [ 5658 1134 1076 2635 7228 689] [ 1927 472 384 361 908 3043]] precision recall f1-score support					
0	0.79	0.95	0.86	299245	
1	0.63	0.46	0.53	83066	
2	0.60	0.42	0.50	52549	
3	0.59	0.41	0.48	34493	
4	0.63	0.39	0.48	18420	
5	0.73	0.43	0.54	7095	
accuracy				0.74	494868
macro avg	0.66	0.51	0.57	494868	
weighted avg	0.72	0.74	0.72	494868	
Accuracy: 0.7436750810317094 Precision: 0.7232566901554951 Recall: 0.7436750810317094 F1 Score: 0.7225073834621258 Cohen Kappa Score: 0.5159807822270135					



Accuracy: 0.7436750810317094  
Precision: 0.7232566901554951  
Recall: 0.7436750810317094  
F1 Score: 0.7225073834621258  
Cohen Kappa Score: 0.5159807822270135

# Random Forest Algorithm was our chosen model for prediction based on accuracy.

# Tools



1. PYTHON PANDAS
2. PYTHON MATPLOTLIB
3. SEABORN
4. NUMPY
5. HTML/CSS/BOOTSTRAP
6. PYTHON FLASK POWERED API
7. SCIKIT-LEARN AND PICKLE
8. GZIP
9. GITHUB AND HEROKU
10. AWS

# Limitations

## Size of train\_timeseries.csv(16 years of daily data(more than 2mil rows)).

- Capacity of computer was an issue(more than 32 GB run)
- Time Constraint( more than 3 hrs to develop and run model)

## Model('pkl' file of size 6 GB).

- As Github's capacity was 4 GB, compressing of pkl file(600 MB) was done (3 hrs to run)
- Deployment to Heroku was a problem(supported size of 500 MB)
- Alternative cloud platform was AWS(limited time)

Thank you.